# Hadoop 平台的微博热点事件挖掘

谢思发[1],林 琛[1,2],苏 旋[3],江 弋[1]

[1](厦门大学 信息科学与技术学院,厦门 361000)
[2](厦门大学 深圳研究院,广东 深圳 518000)
[3](仟首网络科技有限公司,上海 200000)
E-mail: chenlin@xmu.edu.cn

**摘 要:** 微博作为一种新兴的网络社交服务,其即时通讯功能强大,用户可利用各种手段在微博上实时、快捷地发布社会热点事件. 但是微博平台在短时间内发布大量信息的特点在一定程度上造成了信息的碎片化,而且迅速的信息更新速度易造成重要信息的不易检索. 本文采用 Hadoop 平台,利用其在大数据挖掘方面的优势,提出挖掘微博中热点词的分布式算法,提取热点词组织热点事件,方便用户查询. 此外提出了线性时间复杂度的检测算法,检测热点事件的爆发时间段. 文中采用 Twitter 和新浪微博上的数据集作为测试样本,进行了大量的实验,实验结果表明本文算法能有效的提取微博中的热点事件.

**关 键 词:** 微博; Hadoop; 分布式; 热点事件
**中图分类号:** TP18 **文献标识码:** A **文 章 编 号:** 1000-1220(2014)04-0797-05

## Mining Hot Event from Microblog with Hadoop

XIE Si-fa[1], LIN Chen[1,2], SU Xuan[3], JIANG Yi[1]

[1](School of Information Science and Technology, Xiamen University, Xiamen 361005, China)
[2](Shenzhen Research Institute of Xiamen University, Shenzhen 518000, China)
[3](Channal trans Network (Shanghai) Co., Shanghai 20000, China)

**Abstract:** As a newly emerging social-networking service, Microblog has a strong immediate communication function and can release hot issues of society rapidly by various methods. However, the huge mass of data releasing in a short time leads to the fragmentation of information to some extent. Moreover, the quick updating of information results in the difficulty of retrieving essential issues. In this paper, we propose a distributed algorithm of mining hot spots from Microblog data based on Hadoop, which is superior in big data mining, and detect hot issues according to the extracted spots for users' searching convenience. Furthermore, we put forward the detecting algorithm with a linear time complexity, detecting the time period of the burst of the hot issues. The experiments on Twitter and Sina Weibo show that our algorithm can extract hot issues from microblog effectively.

**Key words:** microblog; hadoop; distributed; hot event

## 1 引 言

微博作为一种新兴的开放式互联网社交服务,有以下特点:传播受众群体的广泛性;传播途径的草根性以及传播内容的原创性. 其即时通讯功能十分强大,可利用各种手段实时、快捷、现场发布大的突发事件或引起全球关注的大事. 然而,微博平台短时间内发布较大量信息的特点在一定程度上造成了信息的碎片化,导致过多"耗费"时间,而非"使用",并且迅速的信息更新速度易造成重要信息的不易检索.

热点话题发现与跟踪(topic detection and tracking, TDT)[1-3]就是从网络文本集中识别出突发性热点话题,并跟踪话题的演变过程. 已有很多的算法被提出用来实现 TDT,如 Allan 等[4]利用单路径聚类算法,结合一个新闻阈值模型实现了一个在线新闻监测系统;吴永辉等[5]提出的基于主题的自适应、在线网络热点发现方法及新闻推荐系统;此外 Agarwal 等[6]研究了在高动态环境中利用图算法实时地挖掘社会事件. 但是传统的 TDT 在微博这种短文本处理效果不理想.

推特、微博的出现及普及,促进了学者对海量微博信息的数据挖掘研究工作,并取得了一定的进展[7,8]. 包括微博检索[9],Twitter 上的话题识别[10],特定时间段的 Twitter 总结[11]. 也有部分研究是致力于事件检测[12-15],但是这些研究主要是对新奇事件或特定领域事件的检测,缺少全局观念. 本文采用类 K-Means 聚类算法,对挖掘的热点词进行聚类生成热点事件. 可在特定时间段内,全面地挖掘热点事件.

## 2 热点事件检测

### 2.1 定义

微博标签:每个微博标签 MT 由两部分组成,微博内容 C 和该微博内容发表的时间 T. 记 MT 为(T,C);

收稿日期:2013-01-25 收修改稿日期:2013-03-02 基金项目:国家自然科学基金项目(61102136,61001013)资助;福建省自然科学基金项目(2011J05158)资助;深圳市科技创新基础研究项目(JCYJ20120618155655087)资助. 作者简介:谢思发,男,1989 年生,硕士,研究方向为微博检索;林 琛(通信作者),女,1982 年生,博士,助教,研究方向为 Web 数据挖掘与推荐系统;苏 旋,男,1982 年生,硕士,研究方向为微博营销;江 弋,男,1960 年生,副教授,研究方向为数据挖掘.

单词序列: 单词序列 WS 定义为( W,Fs) . 其中 W 为单词,Fs = { f1,f2,... fn} 是单词 W 的词频序列,每个 fi 对应于一个单位时间内的词频.

爆发度: 我们采用[16]中的定义来做为单词的爆发度. 假定每个单词的词频服从高斯分布,则爆发度 Bi 定义为: $\{fi - \mu - 2\sigma\}$ . 参数 $\mu$ 和 $\sigma$ 可以采用最大似然估计计算.

爆发序列: 爆发序列 BS 定义为( W,Bs) . W 为单词,Bs = { b1,b2,... bn} 是爆发序列,每个 bi 对应一个词频的爆发度.

### 2.2 算法流程

直观上,热点事件有一定规律. 首先其出现的频率必须较高,其次具有爆发性,即在一定时间段里出现的频率会剧增,这段时间即爆发时间段. 据此本文采用如下流程来提取热点词. 先对原始数据进行预处理,提取每条微博的内容和其对应的发表时间生成 MT; 对微博内容进行分词,并统计词语在单位时间内的频率生成 WS. 根据词频计算单词的爆发度生成 BS; 最后采用[17]的聚类算法 WKSC 对热点词进行聚类,生成热点事件. 同时采用一个线性时间复杂度的算法,计算单词的爆发时间段.


图 1 热点事件检测流程图

Fig. 1 Flow char of hot event detecting

### 2.3 WS 和 BS 的生成

原始数据在经过预处理后生成微博标签 MT( T,C) ; 我们将 MT 作为 Map 端的输入,每次读入一条 MT 数据,用中文分词法将 C 分成不同的单词 W. 创建一个时间数组 TL,并初始化为 0,数组长度等于总的时间片段个数.

**算法 1**. 生成 WS 的算法描述
**输入:** 微博标签 MT
**输出:** 单词序列 WS
**定义:** 单词数组 WL,用来保存每条 MT 产生的单词;
时间数组 TL,用来记录单词的词频.

```
Map 端:
1. WL←IKAnalyzer( MT. c) ;
2. For i = 1 to WL. Length do
3. Init( TL) ;
4. j←GetIndex( MT. t) ;
5. TL [j] ←1;
6. return ( WL [i] ,TL) ;
7. end for
Reduce 端:
8. Init( TL) ;
9. for i = 1 to the count of value do
10. TL + = valuei;
11. return( key,TL) ;
12. end for
```

根据 T 计算数组下标 j,令 TL [j] = 1; 然后将每个单词 W 作为 key,数组 TL 作为 value 输出到 Reduce 端. 在 Map 端,将每个 key 的 value 值相加,得到一个总的 svalue. 然后将 key 和 svalue 作为键值对返回. 经过这样处理就能得到 WS.

得到 WS 后,我们仍然利用 Map-Reduce 处理 WS 来生成 BS. 我们在 Map 端读入一条 WS 数据后,对于每个 ti 时间段的词频 fi,将其作为 value 值分别发给 ti 后的 W 个时间段,而 key 是由单词 w 和对应的要发给的时间段 j,以及在时间段 j 的词频 fj 组成的字符串. 这样在 Reduce 端,除最前面的 W 个时间片段外,每个时间片段都能得到其前面的 W 个词频. 从而得到该时间片段的爆发值. 最后再用一步 MapReduce,将相同单词的 BL 数组相加得到每个单词的爆发序列.

**算法 2**. 生成 BS 的算法描述
**输入:** 单词序列 WS
**输出:** 爆发序列 BS
**定义:** TL 为词频数组; W 为滑动窗口的长度;
BL 为爆发度数组 Map 端:

```
1. for i = 1 to TL. Length do
2.    for j = i + 1 to i + w do
3.    return ( key j TL [j] ,TL [j]) ;
4.    end for
5. end for
Reduce 端:
6. Init( BL)
7. $\mu \leftarrow \sum_{i=1}^{w} value_i$
8. $\sigma^2 \leftarrow \sum_{i=1}^{w} (value_i - \mu)^2$
9. BL [j] ←TL [j] $-\mu-2\sigma$
10. return( key,BL) .
```

### 2.4 热点事件提取及爆发时间段检测

本文采用 WKSC[17]时间序列聚类算法,将热点词进行聚类,生成热点事件. WKSC 算法步骤如下: 采用 Haar 小波变换将高维的时间序列进行降维: 每次取高维时间序列中,相邻时间点的平均值作为新的时间点,以此构成低维的时间序列. 算法首先在低维数据上聚类,并计算每个类的中心点. 然后将时间序列重新归类,并更新每个的类的中心. 最后用低维聚类得到的中心作为高维的聚类的初始中心,如此迭代. 当遇到以下两种情况时,停止迭代: 算法运行到指定的反小波变化层次; 低层时间序列的聚类情况在高层聚类时没发生改变.

另外本文采用了一个线性时间复杂度的算法检测热点词的爆发时间段. 算法的数据结构中,有一个动态的 list L,L 里面的每个元素是一个候选的时间段 T,这个时间段有 4 个元素,元素 st 代表这个时间段开始的时间,元素 et 代表这个时间段结束的时间,Ls 代表这个时间段开始前,不包括时间段开始的那个时间点总共的爆发分数, Rs 代表时间段结束后,包括时间段结束的那个时间点总共的爆发分数,另外有一个变量 Cs,计算到当前为止所有的爆发分数之和.

**算法 3**. 爆发时间段检测
**输入:** 热点词的爆发序列 BL
**输出:** 爆发片段 L

```
1. Init( L) //初始化 L 为空
2. for i = 1 to BL. Length
3. Cs´←Cs
4. Cs←Cs + BL [i]
5. If ( BL [i] <0)
6. Continue
7. Temp←{ i,i,Cs´,Cs}
8. while( true) do
9. Merge←null
10. for I = L [L. Length] to L [1] do
```

```
11. if ( I. Ls < Temp. Ls)
12. Merge←I//如果某个时间段的开始前分数 < Cs′,那么记录下这个
    时间段,退出搜索
13. break
14. end if
15. end for
16. if( Merge = = null ‖ Merge! = null&&Merge. Rs > Temp. Rs)
17. L. Add( Temp)
18. break
19. Temp. st = Merge. st
20. Temp. Ls = Merge. Ls
21. Delete( Merge)
22. end if
23. end for while
24. end if
```

## 3　实验比较与分析

实验共使用 2 个数据集,第一个数据集是 2011 年 1 月 23 号到 2 月 8 号在 Twitter 上发表的说说,共 2. 2G,我们取一天为单位. 第二个数据是新浪上的微博,时间从 2009 年 8 月到 2012 年 5 月,共 3. 3G,我们取一个月为单位. 实验所用的 Hadoop 集群配置如下: Hadoop 版本为 1. 0. 1,共有 8 个节点. 每个节点的 CPU 为 4 核 8 线程的 Inter( R) Core( TM) i7,每个节点的硬盘大小为 1T. 其中主节点的内存为 64G,从节点的内存为 32G.

我们分别采用了传统的单节点处理方法和本文提出的基于 Hadoop 的方法,对两组数据进行单词序列和爆发序列的统计,并做了时间性能上的比较. 同时也分析了不同 Hadoop 处理节点个数对时间性能的影响,如表 1 所示. 在传统单节点处理方法中,由于受内存的限制,需对数据进行分段处理. 这导致了过多的 I/O 操作和数据查找操作,从而引起大量的时间开销. 从表 1 可以看出,相比于传统单节点方法,Hadoop 的分布式方法在时间性能上有很大的优势. 另外增加节点数量并没有很好的改进时间性能,原因是 Hadoop 集群的初始化需要一定的时间,这部分时间是不可减少的. 但新浪数据在增加节点时的性能改进要好于 Twitter 数据,这是因为新浪的数据较大,发挥了 Hadoop 集群在处理大数据方面的优势. 所以如果处理的数据更大,增加处理节点对时间性能的改进将更明显.

表 1　不同处理节点的时间性能

Table 1　Time performance of different processing nodes

| 节点 | Twittter 数据集 | 新浪数据集 |
|---|---|---|
| 1 处理节点 | 416s | 1322s |
| 4 处理节点 | 405s | 1199s |
| 8 处理节点 | 384s | 1069s |
| 传统单节点 | 3900s | 9700s |

我们分别对两组数据做爆发度的时序图,图 2-图 4( 见下页) 是从新浪微博中挖掘的热点事件的时序图,下页图 5 和图 6 是从 Twitter 中挖掘的热点事件时序图.

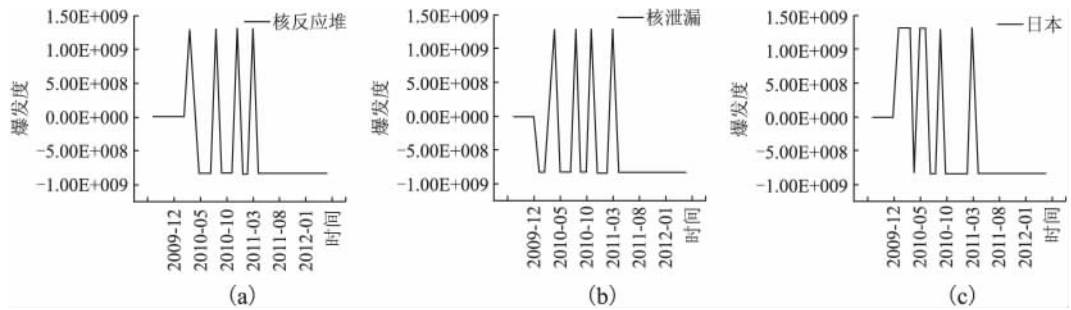


图 2　日本核泄漏

Fig. 2　Japans nuclear leak

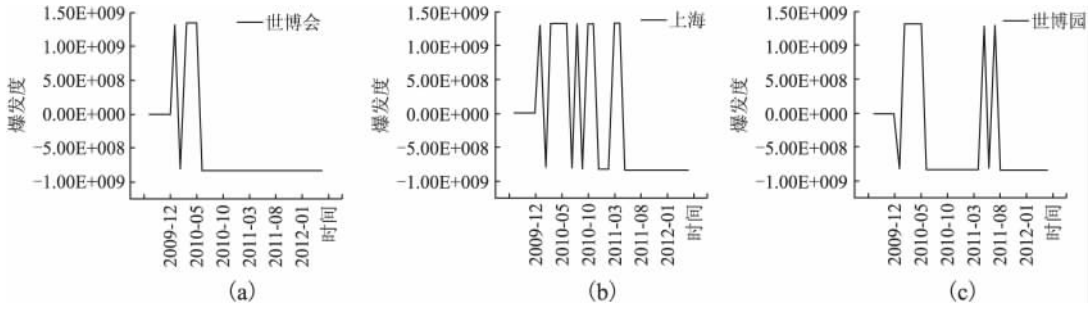


图 3　上海世博会

Fig. 3　Shanghai′s World Expo

从图 2-图 6 可以看出,与同一热点事件相关的热点词具有相同趋势的时序图. 用 WKSC 算法可有效的聚类热点词,生成热点事件. 用算法 3 得到的各热点事件的爆发时间与事件的发生时间对比如下页表 2. 在每个热点事件中,部分单词会存在噪声,如上海世博会中的上海、埃及暴动中的 rally. 这些单词比较常用,所以有些爆发点是与其他事件相关的. 在提

取爆发时间段时,我们是用每个热点事件中,爆发时间段较少的单词来进行计算的,以此来减少噪声的影响. 从表2看出,

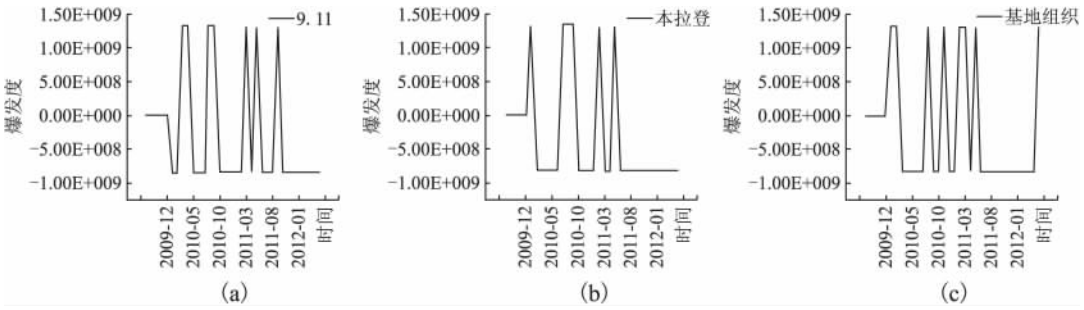


图4 本·拉登被击毙

Fig. 4 Bin Laden is shoot dead

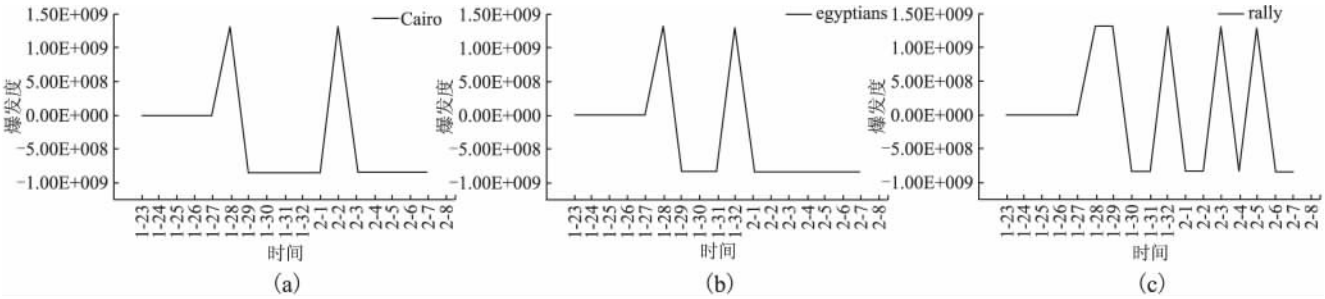


图5 埃及暴动

Fig. 5 Egypt riot

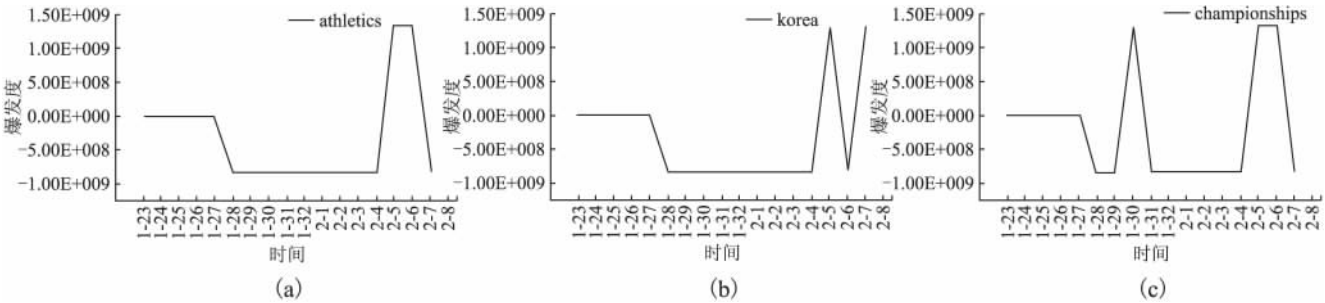


图6 韩国举行世界田径锦标赛

Fig. 6 Korea's world athletics championship

表2 各热点事件爆发时间段检测

Table 2 Burst time of hot event

| 编号 | 热点事件 | 相关时间 | 检测的爆发时间段 |
|---|---|---|---|
| 1 | 日本核泄漏 | 2011 年 3 月 | 2010 年 3 月、2010 年 8 月、2010 年 12 月、2011 年 3 月 |
| 2 | 上海世博会 | 2011 年 5 月 | 2011 年 1 月至 2011 年 5 月 |
| 3 | 本·拉登被击毙 | 2011 年 5 月 | 2010 年 1 月、2010 年 7 月至 2010 年 9 月、2011 年 2 月、2011 年 5 月 |
| 4 | 埃及暴动 | 1 月 28 号、2 月 3 号 | 1 月 28 号、2 月 3 号 |
| 5 | 韩国举行世锦赛 | 1 月 29 号、2 月 6 号 | 1 月 29 号、2 月 6 号至 2 月 7 号 |

对于小单位时间的 Twitter 数据,算法 3 可以准确地检测事件的爆发时间点. 而对于单位时间较大的新浪数据,则可以有效地跟踪与事件相关的其他事件. 如本·拉登被击毙是在 2011 年 5 月,而在 2010 年 1 月美国方面公布了最新的本·拉登合成照,2010 年 7 月《本·拉登传》出版.

## 4 结论与总结

微博作为新兴的开放式互联网社交平台,其传播消息具有即时量大的特点. 但文本内容短小,消息呈碎片化. 本文采用 Hadoop 框架,利用其分布式处理数据的优点,提出了挖掘微博热点事件的有效算法. 我们采用 Twitter 和新浪微博两组数据进行实验,分别采用不同的时间单位来分析数据. 实验结果表明,算法可有效地检测热点事件,并准确地跟踪事件的发生时间. 以单词为基础挖掘热点事件会忽视一定的上下文语义信息,因此下一步我们将研究 Hadoop 结合 bigram、trigram 或短语抽取等方法来挖掘热点事件.

**References:**

[1] Li Hong, Wei Jin-feng. Netnews bursty hot topic detection based on butsty feature [C]. Proceedings of International Confernece on E-Business and E-Government, Washington DC, USA: IEEE, 2010: 1437-1440.

[2] Holz F, Teresniak S. Towards automatic detection and tracking of topic change [M]. Computational Linguistic and Intelligent Text, Berlin, Germany: Springer-Verlag, 2010: 327-339.

[3] Jing Qiu, Liao Le-jian, Dong Xiu-jie. Topic detcetion and tracking for Chinese news web pages [C]. Proceedings of Seventh International Conference on Advanced Language Processing and Web Information Technology, Washington DC, USA: IEEE Computer Society, 2008: 114-120.

[4] Allan J, Papka R, Lavrenko V. On-line new event detection and tracking [C]. Sigir 98, Proceedings of 21th ACM SIGIR International Conference on Research and Development in Information Retrieval. New York: ACM, 1998: 37-45.

[5] Wu Yong-hui, Wang Xiao-long, Ding Yu-xin, et al. Adaptive on-line web topic detection method for web news recommendation system [J]. Acta Electronica Sinica, 2010, 38(11): 2620-2624.

[6] Manoj K Agarwal, Krithi Ramamritham, Manish Bhide. Real time discovery of dense clusters in highly dynamic graphs: identifying real world events in highly dynamic environments [C]. Proceedings of the VLDB EndowmentVery Large Data Base Endowment Inc (VLDB), 2012, 5(10): 980-991.

[7] Lin Chen, Lin Chun, Li Jing-xuan, et al. Generating event storyline from microblogs [C]. Proceedings of the 21st ACM Conference on Information and Knowledge Management (CIKM), 2012: 175-184.

[8] Sasa Petrovic, Miles Osborne, Victor Lavrenko. Streaming first story detection with application to twitter [C]. The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL), 2010: 181-189.

[9] Efron M. Information search and retrieval in microblogs [J]. Journal of the American Society for Information Science and Technology, June 2011, 62(6): 996-1008.

[10] Mathioudakis M, Koudas N. Twittermonitor: trend detection over the twitter stream [C]. Proceedings of the 2010 International Conference on Management of Data (SIGMOD 2010), New York: ACM, 2010: 1155-1158.

[11] Takamura H, Yokono H, Okumura M. Summarizing a document stream [M]. Advances in Information Retrieval, Springer Berlin Heidelberg, 2011: 177-188.

[12] Sakaki T, Okazaki M, Matsuo Y. Earthquake shakes twitter users: real-time event detection by social sensors [C]. Proceedings of the 19th International Conference on World Wide Web (WWW 2010), 2010: 851-860.

[13] Shamma D A, Kennedy L, Churchill E F. Peaks and persistence: modeling the shape of microblog conversations [C]. Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work (CSCW '11), 2011: 355-358.

[14] Li Jin, Zhang Hua, Wu Hao-xiong, et al. BTopicMiner: domain-specific topic mining system for Chinese microblog [J]. Journal of Computer Applications, 2012, 32(8): 2346-2349.

[15] Weng Jian-shu, Bu-Sung Lee. Event detection in twitter [C]. In Proceedings of the Fifth Annual Conference on Weblogs and Social Media (ICWSM 2011), 2011: 401-408.

[16] Yao Jun-jie, Cui Bin, Huang Yu-xin, et al. Bursty event detection from collaborative tags [C]. World Wide Web (2012), 2012, 15: 171-195.

[17] Han Zhong-ming, Chen Ni, Le Jia-jin, et al. An efficient and effective clustering algorithm for time series of hot topic [J]. Chinese Journal of Computers, 2012, 35(11): 2337-2347.

**附中文参考文献:**

[5] 吴永辉,王晓龙,丁宇新,等. 基于主题的自适应在线网络热点发现方法及新闻推荐系统 [J]. 电子学报, 2010, 38(11): 2620-2624.

[14] 李 劲,张 华,吴浩雄,向军. 基于特定领域的中文微博热点话题挖掘系统 BTopicMiner [J]. 计算机应用, 2012, 32(8): 2346-2349.

[17] 韩忠明,陈 妮,乐嘉锦,等. 面向热点话题时间序列的有效聚类算法研究 [J]. 计算机学报, 2012, 35(11): 2337-2347.